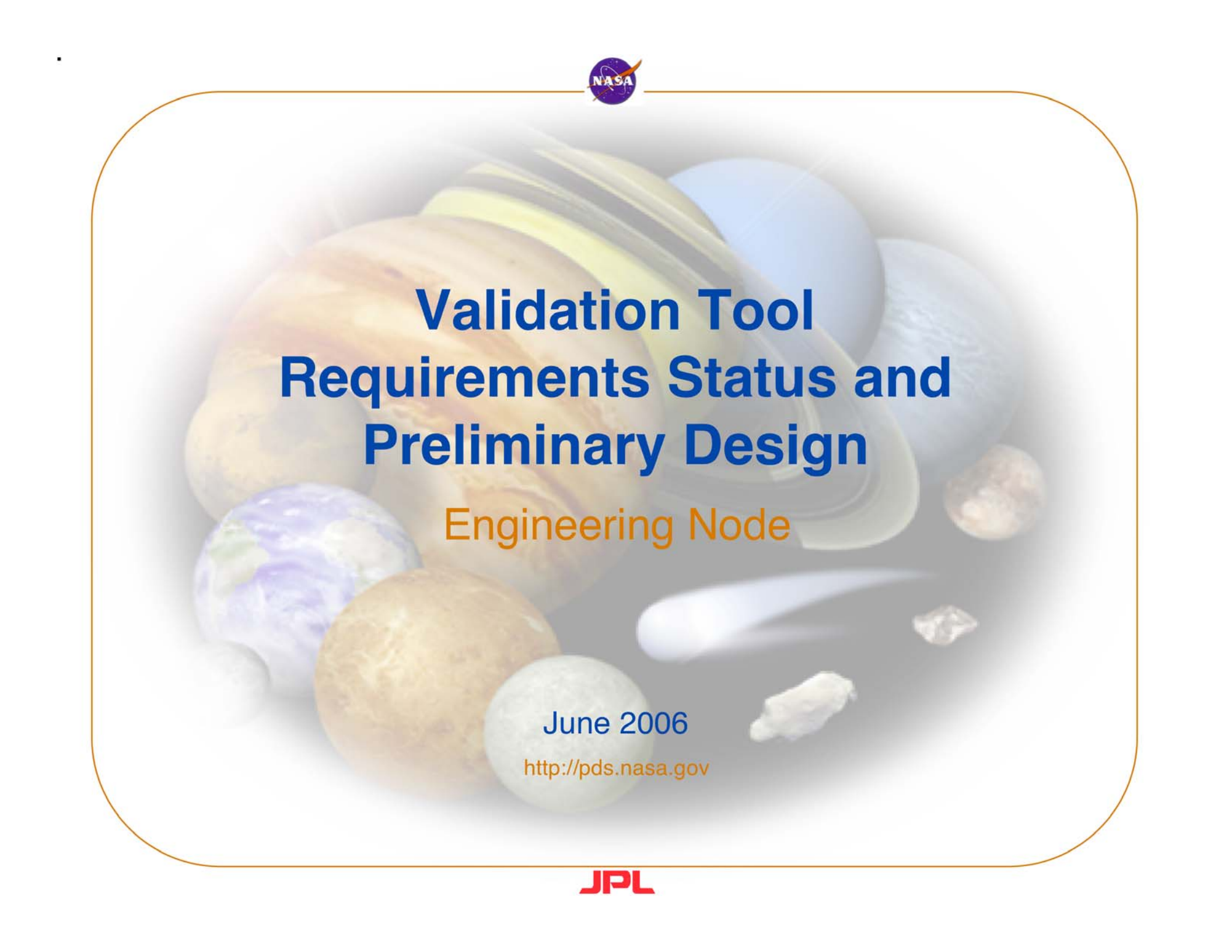


Validation Tool Requirements Status and Preliminary Design

Engineering Node

June 2006

http://pds.nasa.gov

JPL

Topics

- Agenda
- Requirements Status
  - Overview
  - RFA Status
  - Issues
- Preliminary Design
  - Overview
  - Supporting Technology
  - Architecture
  - Software Design
  - Interfaces
  - Test Plan
  - Configuration Management
  - Schedule
  - References

Agenda

9:00 - 9:05 AM	**Opening Remarks**	Dan Crichton (chair) Ed Grayzeck (timekeeper)
9:05 - 9:45 AM	**Requirements/RFA Status**	Sean Hardman
9:45 - 10:30 AM	**Preliminary Design**	Sean Hardman Paul Ramirez
10:30 - 10:45 AM	**Test Plan and Schedule**	Emily Law
10:45 - 11:00 AM	**Action Items, Questions**	Dan Crichton

Requirements Status

Overview

- Version 1.2 (4/1/06) is still the latest version of the Validation Tool Requirements document.
- One of the reasons for holding this telecon is to try and resolve a number of issues with the RFAs.
  - This will aide in the preparation of the next version of the requirements document.
- RFA resolution focus has been on syntactic and semantic validation requirements.
  - RFAs related to content (object) validation requirements have been left open (unaddressed) for the most part.

RFA Status

- Requests for Action (RFA) Process
  - Collect and summarize RFAs (Open)
  - Develop appropriate action (Addressed or Tabled)
  - RFA author approves of action (Closed)
- 88 Total (60+ Distinct)
  - Status
    - 52 Open
    - 04 Addressed
    - 02 Tabled
    - 30 Closed

Issues

- Checksum Inclusion
- Dictionary Creation
- CAT File Value Extraction
- Validating Partial Labels
- PSDD Version Determination
- PDS Version Support
- Requirements Scope

Issue
Checksum Inclusion

- The issue is whether or not to include the checksum related requirements in the document.
- The requirements are L5.VAL.FR.5, L5.VAL.FR.6, L5.VAL.FR.7 and L5.VAL.FR.8. See next slide for details.
- The related RFAs are MG01, RAS08, SS15 and TK25. See the slide after next for details. My summary follows:
  - Susie and Mitch preferred not specifically stating “MD5”. Addressed in version 1.2.
  - Mitch would like the current requirements retained, including the two that I deleted. I deleted the two requirements related to SCR 3-1035 since that SCR was tabled.
  - Dick would like the current set of requirements scraped and rewritten to be more generic.
  - Todd doesn’t believe that these requirements are relevant to the tool anymore.
- Given all of this, and the recent e-mail traffic regarding the SCR 3-1034, I would like to pull them from the document, for now.

Issue
Checksum Inclusion - Related Requirements

L5.VAL.FR.5 - The Tool shall have the capability to calculate a PDS approved checksum against the contents of a file. (L4.VAL.FR.1.2)

Design and implementation of this requirement is subject to the outcome of SCR 3-1034.

L5.VAL.FR.6 - [Deleted] The Tool shall have the capability to calculate a PDS approved checksum against the contents of each data object contained in an data product. (L4.VAL.FR.1.2)

L5.VAL.FR.7 - The Tool shall have the capability to compare and report differences between a PDS approved checksum value referenced in a PDS label and a PDS approved checksum calculated against the contents of the referenced file. (L4.VAL.FR.1.2)

Design and implementation of this requirement is subject to the outcome of SCR 3-1034.

L5.VAL.FR.8 - [Deleted] The Tool shall have the capability to compare and report differences between the PDS approved “data object” checksum value referenced in the PDS label and the PDS approved checksum calculated against the contents of each data object described by the PDS label. (L4.VAL.FR.1.2)

Issue
Checksum Inclusion - Related RFAs

RFA #	Submitter	Concern	Recommendation	Action
MG01	Mitch Gordon	MD5 has not yet been accepted as the PDS wide checksum option of choice.	Throughout, replace “MD5” with “the PDS approved”	Open 23-Feb-06 Related to RAS08 and SS15. Addressed 26-Feb-06 Deleted requirements L5.VAL.FR.6 and L5.VAL.FR.8 since both were related to withdrawn SCR 3-1035. Reworded requirements L5.VAL.FR.5 and L5.VAL.FR.7, replacing "MD5" with "PDS approved" and removed the verbage limiting the checksum capability to just data product files. Open 5-Apr-06 Not accepted by M. Gordon.
RAS08	Dick Simpson	The half page of MD5 requirements based on the presumed outcome from SCR 3-1034 is a misguided attempt to work the requirements process from the bottom up. MD5 can be justified as "useful" in ensuring data integrity; but it's not the sole answer and the	Remove L5.VAL.FR.(5) through L5.VAL.FR.(8) and substitute something more generic, along the lines of "The Tool shall have the capability to calculate an MD5 checksum against the contents of any file."	Open 24-Feb-06 Related to MG01 and SS15. Addressed 26-Feb-06 Deleted requirements L5.VAL.FR.6 and L5.VAL.FR.8 since both were related to withdrawn SCR 3-1035. Reworded requirements L5.VAL.FR.5 and L5.VAL.FR.7, replacing "MD5" with "PDS approved" and removed the verbage limiting the checksum capability to just data product files.
SS15	Susie Slavney	L5.VAL.FR.5: Maybe it is not a good idea to specifically list MD5 for checksums because PDS has not agreed to use MD5 yet.		Open 3-Feb-06 Related to MG01 and RAS08. Addressed 26-Feb-06 Deleted requirements L5.VAL.FR.6 and L5.VAL.FR.8 since both were related to withdrawn SCR 3-1035. Reworded requirements L5.VAL.FR.5 and L5.VAL.FR.7, replacing "MD5" with "PDS approved" and removed the verbage limiting the checksum capability to just data product files.
TK25	Todd King	SCR 3-1034 addresses only checksums which are maintained external to the product. There is no requirement or expectation that checksums will be in the label. Requirements L5.VAL.FR.5 and L5.VAL.FR.7 are no longer relevant	Remove L5.VAL.FR.5 and L5.VAL.FR.7	Open 3-Apr-06 Related to MG01, RAS08 and SS15?

Issue
Dictionary Creation

- Although the dictionary creation related requirements elicited a good response during the review, I have received three RFAs requesting their removal.
- The requirements are L5.VAL.FR.16 and L5.VAL.FR.17. See next slide for details.
- The related RFAs are DT03, RAS16 and TK12. See the slide after next for details. My summary follows:
  - Basically David, Dick and Todd believe that these requirements have no place in a validation tool.
- I tend to agree and suggest that these two requirements be deleted from the document.

Issue
Dictionary Creation - Related Requirements

L5.VAL.FR.16 - The Tool shall be capable of generating data dictionary element definition templates for elements and element values referenced in the PDS label but not in the PSDD. (L4.VAL.FR.2)

The element definition template that is generated will be limited in scope to the information that can be rendered from the PDS label. The data provider will be responsible for supplying the additional information required to complete the element definition.

L5.VAL.FR.17 - The Tool shall be capable of generating a data dictionary, in standard ODL format, of element definitions for elements referenced in the PDS label. (L4.VAL.FR.2)

The data dictionary that is generated will be limited in scope to the set of elements referenced in the PDS label.

Issue
Dictionary Creation - Related RFAs

RFA #	Submitter	Topic	Concern	Recommendation	Action
DT03	David Tarico	Requirement (Remove)	L5.VAL.FR16 and L5.VAL.FR16 Generating data dictionaries and element templates isn't a part of the validation process that I am aware of. L4.VAL.FR.2 is cited as the source of these requirements, but these two items are not extensions of L4.VAL.FR.2 in my opinion.	Remove L5.VAL.FR16 and L5.VAL.FR16[17] or cite another source for their existence, and perhaps give an explanation of what role they serve in the validation process.	Open 4-Apr-06 Related to RAS16 and TK12.
RAS16	Dick Simpson	Requirement (Remove)	L5.VAL.FR.(16) and (17): I see no connection between these and L2. I see no way that data dictionary definitions could be generated on the fly by a "validation" tool. At best these make sense only as requirements for a "generation" tool.	Omit these requirements.	Open 24-Feb-06 Related to DT03 and TK12. Addressed 27-Mar-06 There was quite a bit of discussion regarding these two requirements during the review and the feedback regarding the capabilities they describe was pretty positive. It is my understanding that kwvtool offers some of what is described by L5.VAL.FR.16 and that ddict offers some of what is described by L5.VAL.FR.17. Both of these requirements were slightly reworded as a result of RFA SH02. If the consensus is that these requirements are better suited to a generation tool, then I will pull them from this document. Open 4-Apr-06 Due to related RFAs and Dick's less than resounding acceptance.
TK12	Todd King	Requirement (Remove)	L5.VAL.FR.(17) Why do we want this capability? We should preserve the entire data dictionary, not subsets.		Open 23-Jan-06 Related to DT03 and RAS16. Addressed 27-Mar-06 There was quite a bit of discussion regarding this requirement during the review and the feedback regarding the capability it describes was pretty positive. It is my understanding that ddict offers some of what is described by L5.VAL.FR.17. This requirement was slightly reworded as a result of RFA SH02. Dick pointed out in RFA RAS16 that this requirement may be better suited as a generation tool requirement. If that is the consensus, then I will pull it from this document. Open 4-Apr-06 Not accepted by T. King.

Issue
CAT File Value Extraction

- In RFA MG02, Mitch made the following request:
  - Add a requirement that the tool check all *.CAT files for 'ID's (e.g., DATASET_ID, REFERENCE_KEY_ID, etc.) and add the corresponding values to its list of 'Standard values' for use in the current validation run. Perhaps include these in the log file and also identify which were not already standard values.
- My response was that this request is outside of the current scope of the tool.
- Mitch didn't agree with my response. In addition, Todd and David responded with their support for Mitch's request.
- I would rather see a requirement for building a local data dictionary from the CAT file contents. That dictionary could then be given as input to the Validation Tool. I still think it is out of scope.
- This approach simplifies the tool interface and limits outside dependencies.

Issue
Validating Partial Labels

- The related RFAs are SS14 and TK18. See the next slide for details.
- In response to SS14, the following requirement was added:
  - L5.VAL.FR.23 - The Tool shall include the contents of external files referenced by ^STRUCTURE pointers when validating a PDS label.
- RFA TK18 suggests changing “shall” to “shall be able to”.
  - This seems appropriate since it allows for the validation of a label without including the contents of external files.
- In addition, TK18 requests the following requirement be added:
  - The tool shall be able to validate a label fragment which can be referenced by a structure pointer (^STRUCTURE).
- Should the tool validate partial labels?
- The reason I ask, is because the PDS Standards Reference does not provide the grammar or a description of what constitutes a valid partial label.
- If this requirement is desired, then additional requirements specifying partial label validation should be developed.

Issue
Validating Partial Labels - Related RFAs

RFA #	Submitter	Topic	Concern	Recommendation	Action
SS14	Susie Slavney	Requirement (Add)	The tool should be able to find and use format files as part of the label.		Open 3-Feb-06 Related to TK18 Addressed 1-Apr-06 Requirement L5.VAL.FR.23 (The Tool shall include the contents of external files referenced by ^STRUCTURE pointers when validating a PDS label.) was added to the document. Open 4-Apr-06 Accepted by S. Slavney but RFA TK18 raised additional issues.
TK18	Todd King	Requirement (Modify)	L5.VAL.FR.23: It should be possible to validate a label without including the ^STRUCTURE content. This will permit validation of the unaltered content of a label. It should be possible to independently validate a file containing a label fragment such as a "STRUCTURE" description.	In L5.VAL.FR.23 change "shall" to "shall be able to". Add a new requirement that reads: The tool shall be able to validate a label fragment which can be referenced by a structure pointer (^STRUCTURE).	Open 3-Apr-06 Related to SS14.

Issue
PSDD Version Determination

- The issue here is whether the Validation Tool should determine the appropriate version of PSDD to use for validation.
- The related RFAs are TK07 and TK24. See the next slide for details.
- RFA TK07 suggests the addition of the following requirement:
  - L5.VAL.FR.2.X - The Tool shall use the appropriate version of the PSDD.
- My response was that the Validation Tool should validate its target based on the PSDD specified by the user.
- Todd didn't agree and authored TK24, which requested the addition of a requirement specifying how the tool will determine the appropriate version of the PSDD.
- I still think the user should determine what version of the PSDD to validate against and pass a reference to that file as input to the tool.
  - This greatly simplifies the tool by not requiring the tool to query the PDS catalog or other source to find the latest or appropriate PSDD.
  - It is not clear to me that there is enough information in a PDS label or in the PSDD itself to determine the appropriate version to use.

Issue
PSDD Version Determination - Related RFAs

RFA #	Submitter	Topic	Concern	Recommendation	Action
TK07	Todd King	Requirement (Add)	L5.VAL.FR.(2) There is no requirement that validation is to use the appropriate version of the PSDD. Perhaps a requirement such as: L5.VAL.FR.2.X - The Tool shall use the appropriate version of the PSDD.		Open 23-Jan-06 Related to TK24. Addressed 26-Mar-06 What exactly is the appropriate version of the PSDD for any given execution of the tool? The tool should validate its target based on the PSDD specified. It is up to the user to determine the desired version of the PSDD. Open 4-Apr-06 Not accepted by T. King.
TK24	Todd King	Requirement (Add)	Should the instance of the PSDD used for validation be determined by the value of the PDS_VERSION_ID? Currently the PDS_VERSION_ID does not have sufficient resolution to determine minor releases of the PSDD. For example, the current version is PDS3. Hopefully in the future this will be changed. Even if its not validation can use the most recent version of the major release (which is currently 3.7) when it encounters a version id of PDS3.	Add the requirement that the validation tool shale be able to select the PSDD to use for validation based on the value of the PDS_VERSION_ID element.	Open 3-Apr-06 Related to TK07.

Issue
PDS Version Support

- What version of the PDS Standards should the Validation Tool support?
- It has been suggested (by an unnamed source) that only PDS version 2 and 3 labels should be supported.
- I would like to have a requirement specifying the supported version(s).

Issue
Requirements Scope

- This issue is probably deserving of its own telecon, but if we have time we will try to touch on it at this time.
- Numerous RFAs are related to this issue.
- The level 4 requirements were authored with the intent of providing a transition between the higher-level requirements and the level 5 requirements.
- Ultimately, the level 4 requirements should be contained in another document.
- An attempt has been made to scope the requirements document to correspond with the first release of the tool. Limiting the scope will help to reach a consensus.
- Now that the level 3 requirements have been approved, the level 4 requirements and the trace matrix will be revisited.

Preliminary Design

Overview

- The other reason for holding this telecon is to discuss some design aspects of the Validation Tool.
- The preliminary design has focused on aspects of the Validation Tool not likely to change during requirements finalization.
  - Syntactic and semantic validation based on chapter 12 of the PDS Standards Reference.
  - The structure and content of a PDS label.
  - The format and content of a PDS compliant data dictionary.
- Supporting technologies have been incorporated where possible to reduce the overall effort of building the Validation Tool.

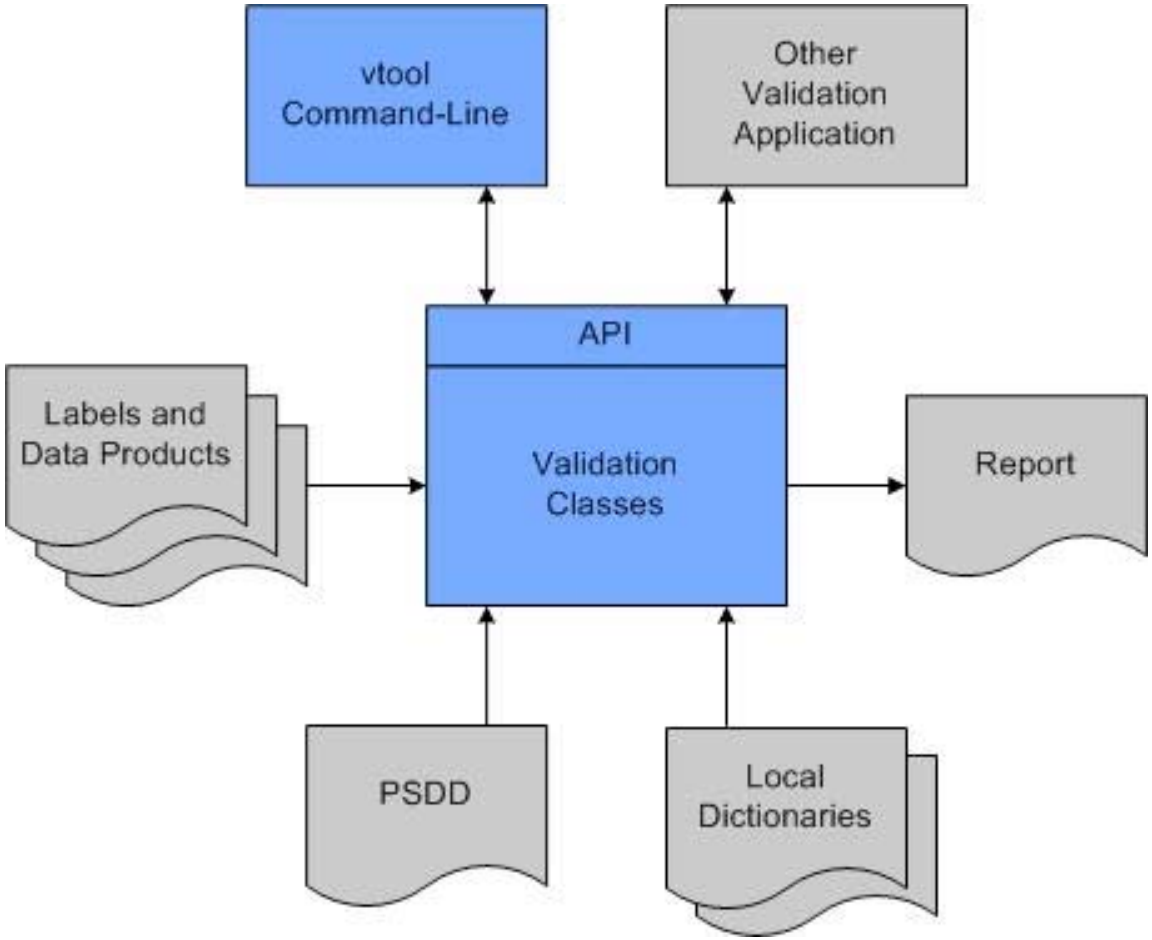
Supporting Technology (1 of 2)

- Java 2 SDK, Standard Edition, Sun Microsystems.
  - Java support is available on the PDS supported platforms (e.g. Linux, Solaris, Windows and Mac OSX).
  - Enables support for additional platforms without additional effort.
  - The tool will be written entirely in Java using version 1.4 for compatibility with older environments.
- Commons Command Line Interface (CLI), Apache Software Foundation.
  - An API for processing command line interfaces.
  - Alleviates the need to develop a command-line parser.
  - Provides for standard and consistent command-line interfaces.
- LOG4J, Apache Software Foundation.
  - An API that allows for control of which log statements are output with arbitrary granularity.
  - Alleviates the need to develop an error message handler.
  - Aides in meeting the Validation Tool's numerous and specific reporting requirements.

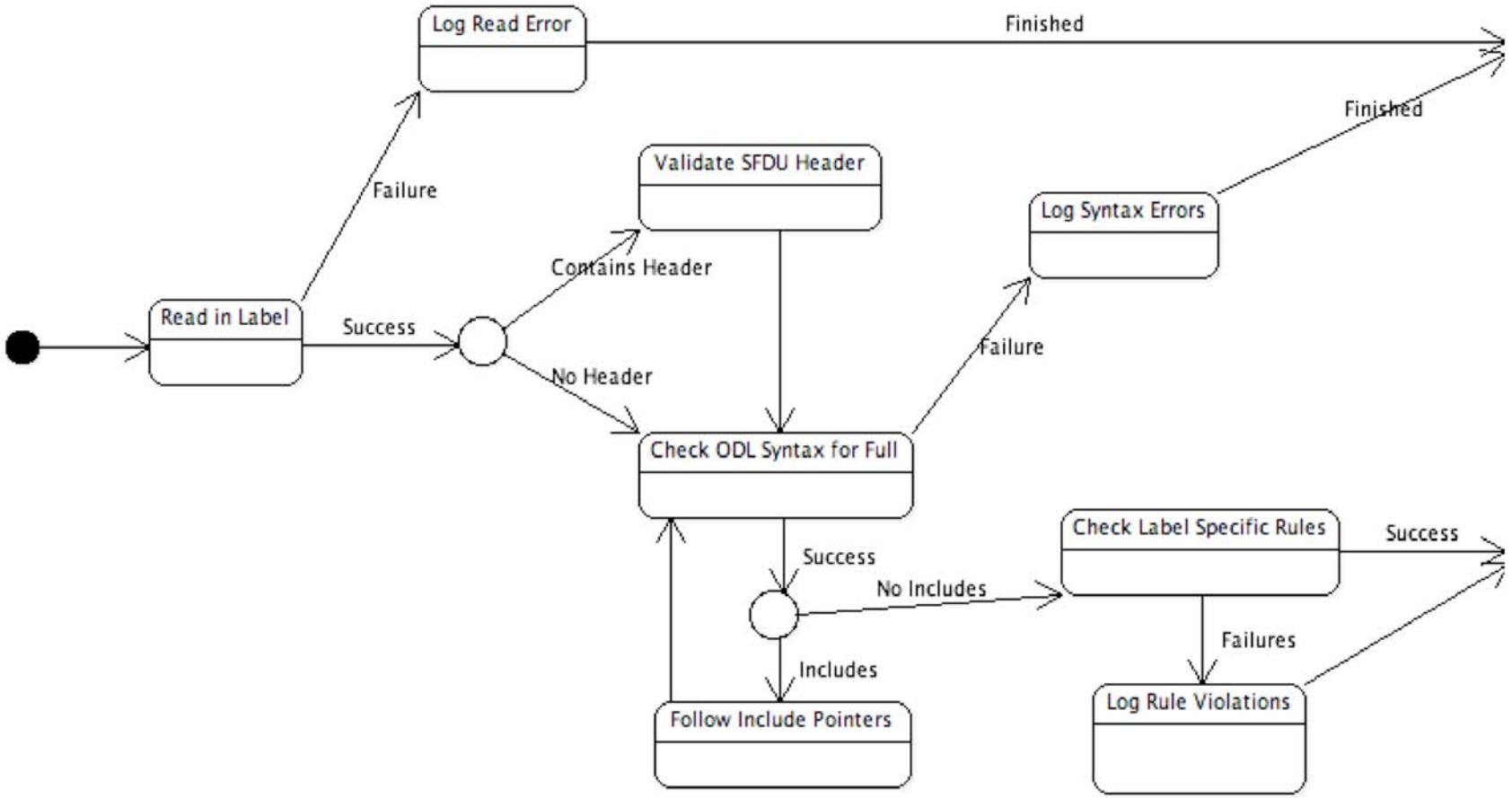
Supporting Technology (2 of 2)

- ANother Tool for Language Recognition (ANTLR), Terence Parr.
  - A language tool that provides a framework for constructing recognizers, compilers, and translators from grammatical descriptions.
  - Alleviates the need to develop a label parser from scratch.
  - Provides a structured environment for developing a parser.
- JUnit, Erich Gamma and Kent Beck.
  - A regression testing framework used by the developer who implements unit tests in Java.
  - More on this when Unit Testing is discussed.
- Maven, Apache Software Foundation.
  - A software project management and comprehension tool that can manage a project's build, reporting and documentation aspects.
  - This tool is essentially a glorified “make” allowing developers to better manage the build process.

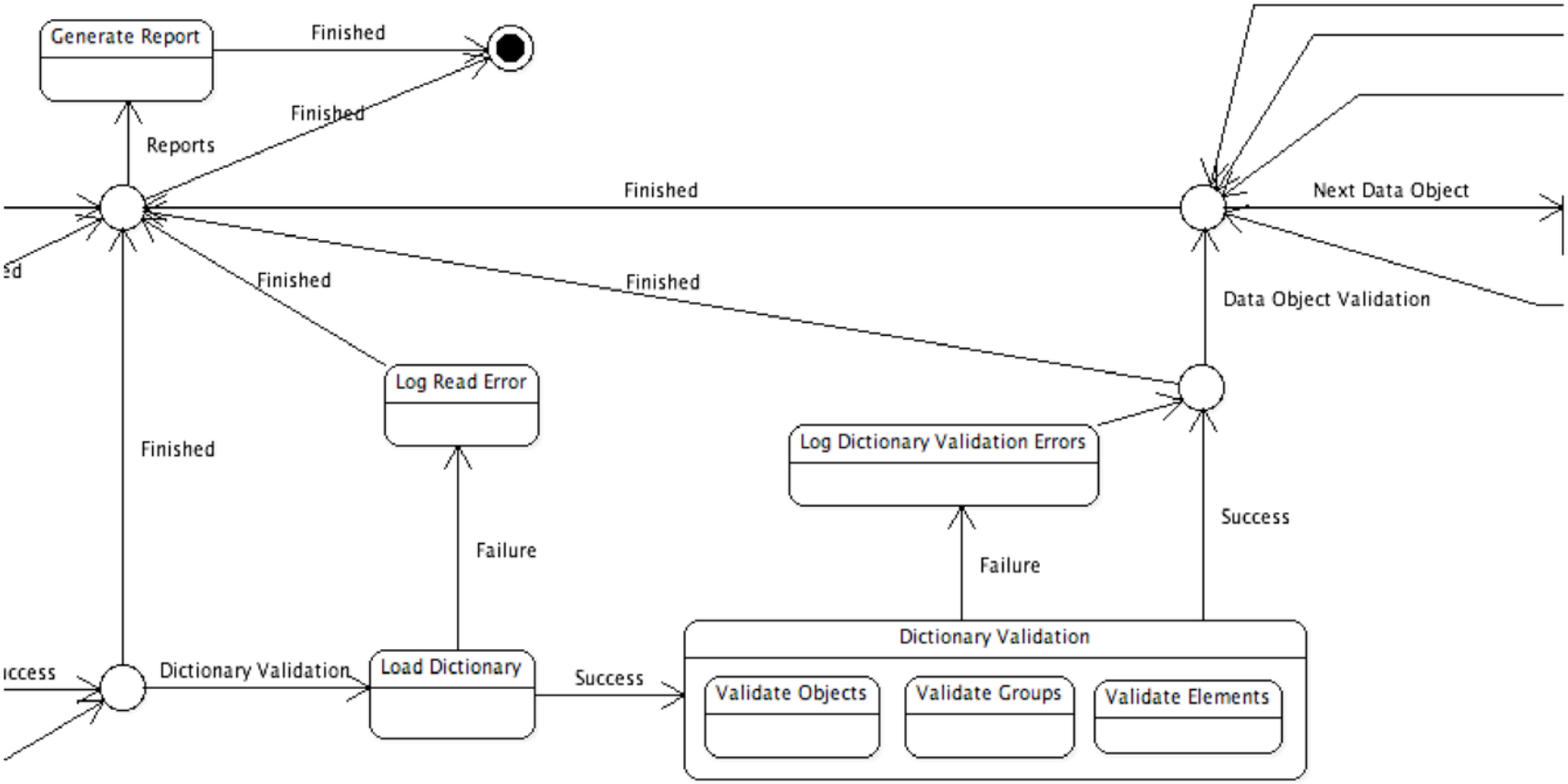
Architecture



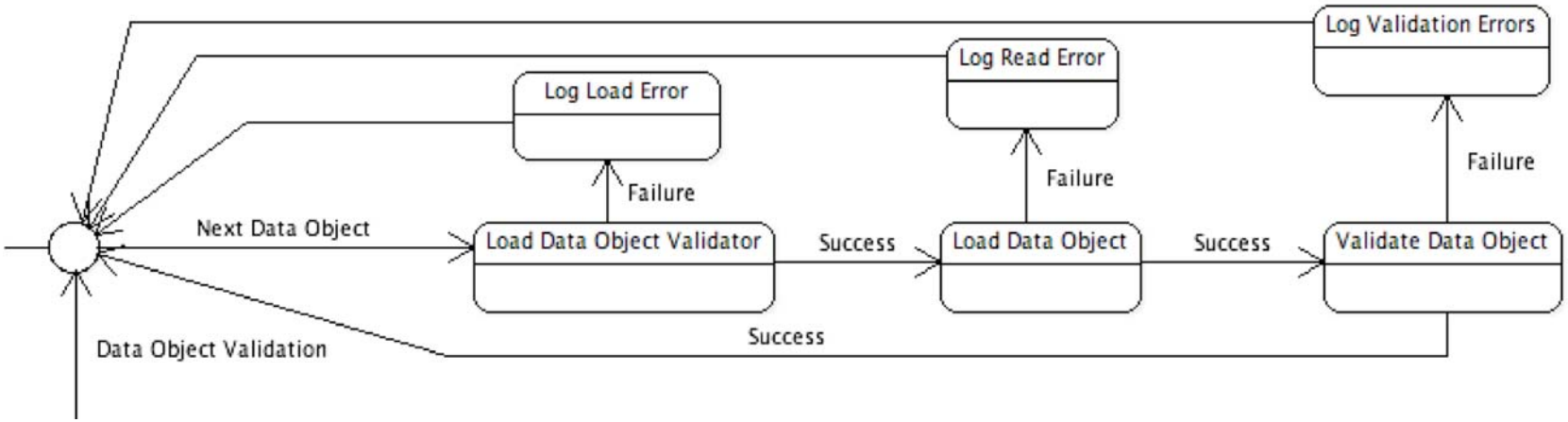
Software Design
State Chart (1 of 3)



Software Design
State Chart (2 of 3)



Software Design
State Chart (3 of 3)



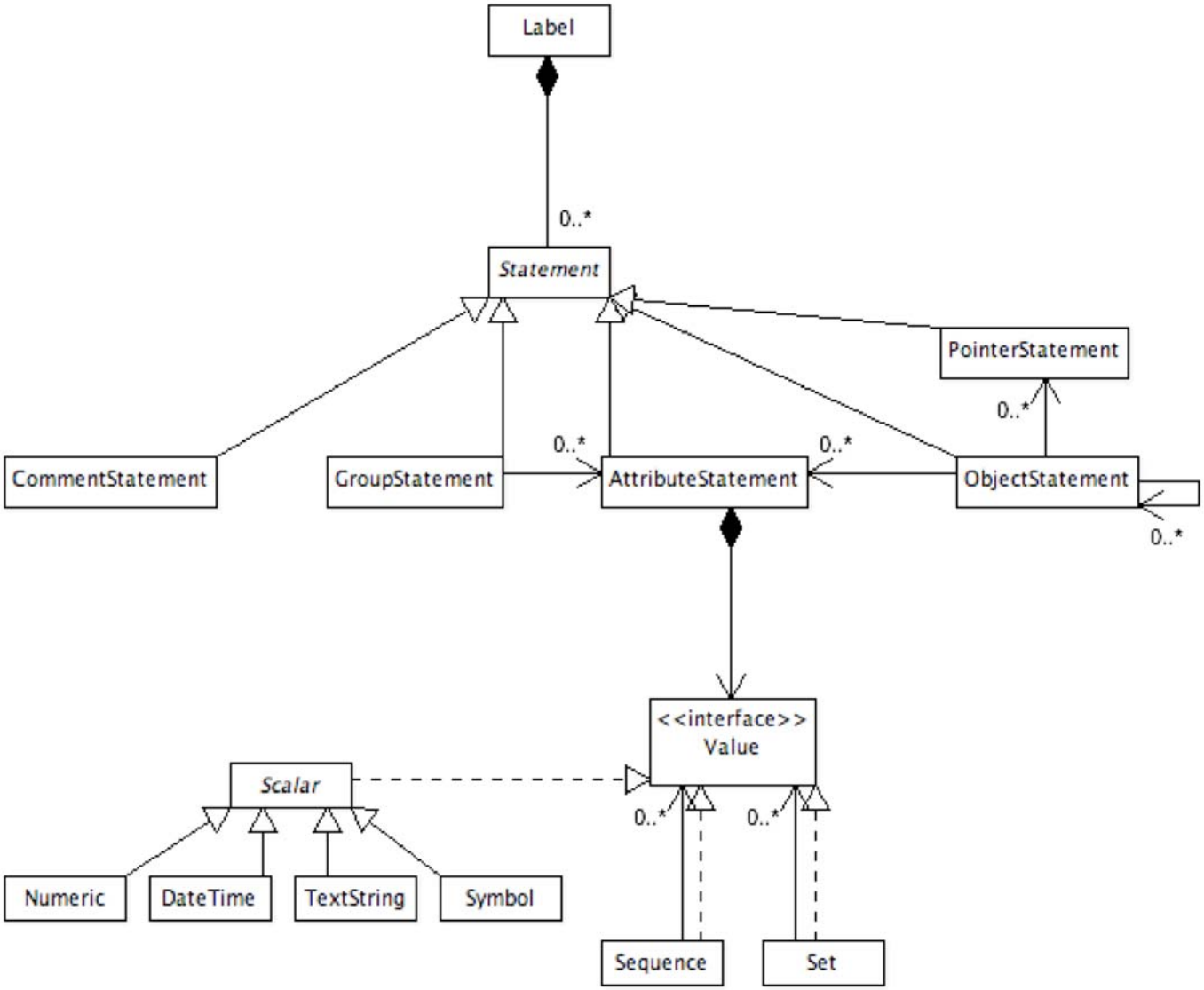
Software Design
State Chart Descriptions (1 of 2)

- Read in Label
  - Locate and read in the specified file.
- Validate SFDU Header
  - If a header is present, validate as specified in chapter 16 of the PDS Standards Reference.
- Check ODL Syntax
  - Validate to the grammar specified in chapter 12 of the PDS Standards Reference.
- Follow Include Pointers
  - Locate and include the contents of files referenced by pointers.
- Check Label Specific Rules
  - If this is a data product label, validate it against the associated rules.
- Load Dictionary
  - Locate and read in the specified dictionary.
  - Multiple dictionaries will be supported in following versions.

Software Design
State Chart Descriptions (2 of 2)

- Dictionary Validation
  - Validate all objects, groups, and elements against loaded dictionary definitions.
- Load Object Validator
  - Depending on the type of object to validate, an appropriate class will be loaded.
- Load Data Object
  - Read in the bytes that make up the data object.
- Validate Data Object
  - Validate the object against its description using the loaded data object validator.
- Generate Report
  - Creates a report according to the specified reporting options.

Software Design
Class Diagram - Label



Software Design
Class Description - Label

- A PDS Label consists of one or more statements.
- Object Statement
  - Example: OBJECT = IMAGE … END_OBJECT = IMAGE
  - May contain Object statements.
  - May contain Pointer statements.
  - May contain Attribute statements.
- Group Statement
  - Example: GROUP = SHUTTER_TIMES … END_GROUP = SHUTTER_TIMES
  - May contain Attribute statements.
- Pointer Statement
  - Example: ^STRUCTURE = “TABLE.FMT”
- Attribute Statement
  - Example: TARGET_NAME = IO
  - Contains a value:
    - Scalar Value (Text String, Numeric, Date/Time or Symbol)
    - Sequence Value
      - May contain multiple Sequences.
    - Set Value
      - May contain multiple Sets.
- Comment Statement
  - Example: /* Image Description */

Interfaces

- Application
  - Command-Line
    - Proposed, obsolete and questionable options are detailed in the following slides.
  - Application Program Interface (API)
    - Under construction.
- Dictionary
  - As specified in the command-line options, one or more PDS compliant data dictionaries can be referenced.
  - This assumes one PSDD and zero to many local data dictionaries.
  - All specified dictionaries will be merged into one master dictionary for validation.
- Report
  - Still working on the report formats.

Interfaces
Application - Command-Line (1 of 3)

- Proposed Options
  - Validation
    - **Target**: Specifies the file(s) to be validated. Accepts multiple entries, directories and wild cards.
    - **Recursive**: Specifies whether specified directories should be traversed recursively. Default is yes.
    - **Ignore Directories**: Specify a text file containing directories to ignore.
    - **Ignore Files**: Specify a text file containing file extensions to ignore.
    - **Follow Pointers**: Specify whether files referenced by pointers are verified for existence and included for parent label validation. Default is yes. The alternative is to validate as partial labels.
    - **Include Directory**: Specify the path to search for pointer files. Default is the current directory.
    - **Dictionary**: Specifies the PDS compliant data dictionary(s) to be referenced for validation. Assumes the full version and not the index. If not provided, dictionary validation is not performed.
    - **Aliases**: Specify whether aliases are allowed. Default is yes.
    - **Data Objects**: Specify whether data objects are validated. Default is yes.

Interfaces
Application - Command-Line (2 of 3)

- Proposed Options (cont)
  - Reporting
    - **Report**: Specifies the report file specification.
    - **Detail**: Specifies report detail (Verbose, Summary or Minimal). Default is Verbose.
    - **Severity**: Specifies the message severity level and above to include (Debug, Info, Warn, Error or Fatal). Default is Info.
    - **Format**: Specifies the format of the report (Human or Machine Readable). Default is Human.
    - **Max Errors**: Specifies the maximum number of errors to report. Default is 300.
  - Miscellaneous
    - **Help**: Display application usage (command-line arguments).
    - **Version**: Display application version.

Interfaces
Application - Command-Line (3 of 3)

- Obsolete Options
  - **-fin**: Specify a text file containing a list of files to validate.
  - **-lef**: Tell lvtool to keep a log file of all files that are skipped and specify a file name for the log.
  - **-nol3d**: Validate files without checking to see if it is a level 3 label.
  - **-nw**: Do not wrap messages. The default is to wrap messages to the next line after it exceeds 72 characters.
  - **-se**: Save temporary expanded label files.
  - **-t**: Direct application output to the terminal.
  - Numerous report related options.
- Option Questions
  - Should the application support progress reporting?
  - Should ^STRUCTURE pointers be treated differently from other pointers?

Test Plan

- Development Testing
  - Unit Testing: Tests performed at the code-level.
  - Integration Testing: Tests performed at the application-level.
- Beta Testing
  - A phased approach involving Node participation.
- Acceptance Testing
  - Final integration testing to be performed for acceptance.
- Documentation
  - Test Plan
    - The plan for testing the Validation Tool will be incorporated into the Engineering Node Test Plan.
  - Test Procedures
    - Procedures will be developed detailing the steps to be performed, test data to be used and the results expected.
  - Test Reports
    - Reports will be prepared for and made available for each release of the Validation Tool.
  - Anomaly Tracking
    - Bug reports will be tracked and reported on at release.

Test Plan
Development Testing - Unit Testing

- The goal of unit testing is to isolate each part of the program and show that the individual parts are correct.
- A test case is developed to test the interface and functionality of a single class.
- Test cases are exercised at build time allowing for immediate detection of coding anomalies.
- Test cases are included with the source code providing a good source of documentation and enabling on-site testing.
- Test cases for the Validation Tool will be built and managed with Junit (testing framework).

Test Plan
Development Testing - Integration Testing

- For now, integration is limited to integration of the classes into a single command-line program.
- A regression test suite will be built supported with documented test data.
  - Each test case will include test data (e.g. PDS label), a description of the scenario being tested and an example report/result.
  - A procedure will be put in place for accepting test cases from the Nodes.
  - Test cases will be captured and managed in the source tree.
  - Where feasible, this test suite will be automated.
- Cross-platform tests will be performed on PDS-supported platforms.
- Installer package tests will be performed ensuring proper installation.

Test Plan
Beta Testing

- Make integration & test build with test procedures available to Node personnel for local testing.
- Implement a phased and iterative approach, initially including three Nodes (ATMOS, GEO and SBN) with increased participation in subsequent phases.
- Provide a method for accepting test cases from the Nodes to be included in the regression test suite.
- Provide a method for tracking bug reports from the Nodes.

Test Plan
Acceptance Testing

- Test cases generated by the Nodes will be fully incorporated into the final regression test suite.
- The final regression test suite will be performed on acceptance build.

Configuration Management

- Source Code Version Control System
  - Source Code (Java, XML, etc.)
  - Unit Test Cases
  - Regression Test Data
  - Online Documentation
  - Open Source Dependencies
- Document Management System
  - Requirements Artifacts
  - Design Artifacts
  - Test Plan, Procedures and Reports
- PDS Web Site
  - Binary Distributions
  - Source Distributions

Schedule

- Obtain MC approval on requirements (Mar 2006)
- Development (Mar - Jun 2006)
  - Design
  - Implementation
  - Unit and Integration Test
  - External Code Review
- Beta Test (Jun 2006)
- Acceptance Test (Jul 2006)
- Obtain MC approval for deployment (Aug 2006)

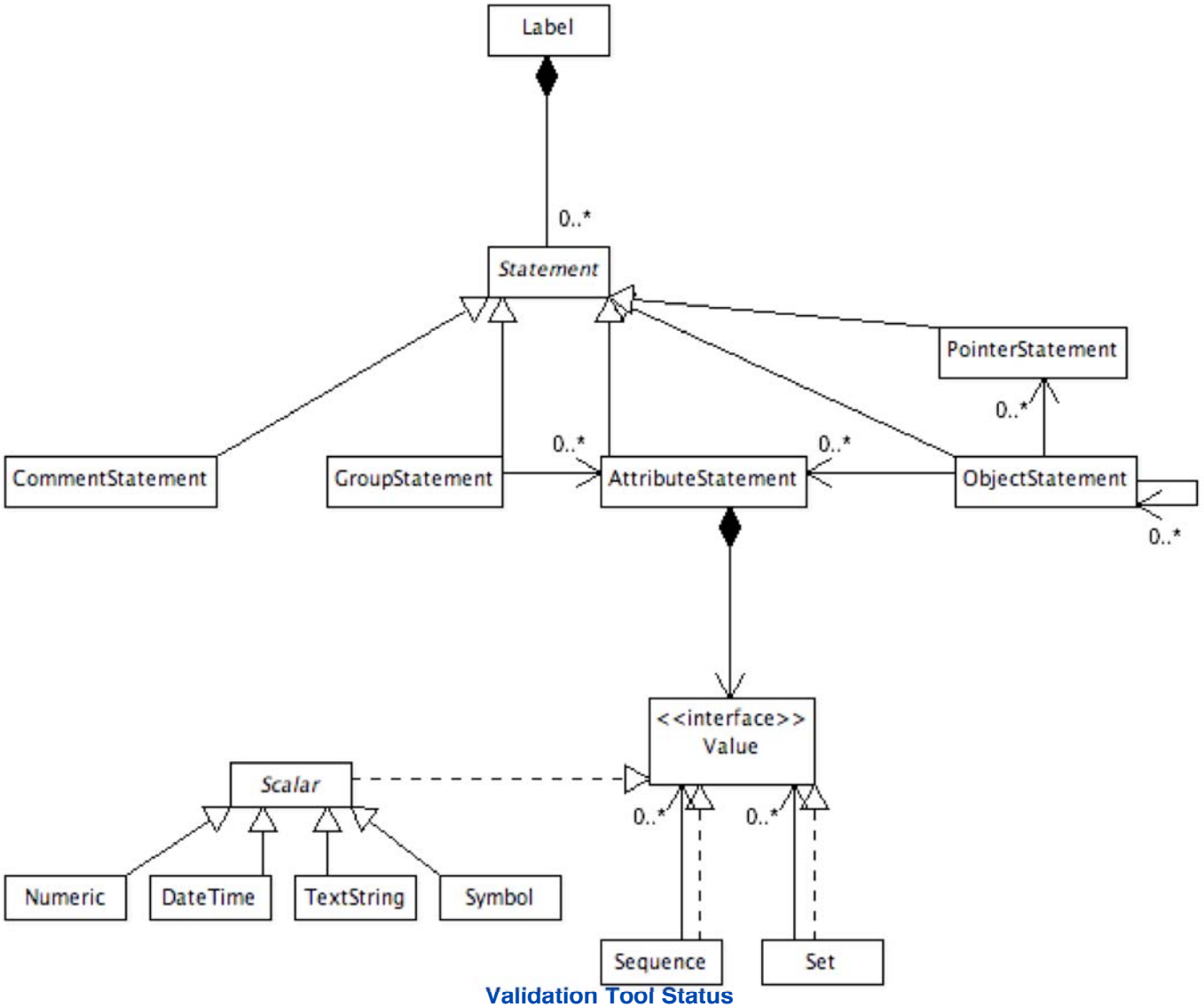
References

- Planetary Data System (PDS) Level 1 and 2 Requirements, March 2, 2006.
- Level 3 Requirement, “1.5.3 PDS will develop, distribute, and maintain tools to assist data producers in validating PDS archival products against PDS Standards.”, March 2, 2006.
- Planetary Data System (PDS) Validation Tool Requirements, April 1, 2006, Version 1.2.
- Planetary Data System (PDS) Standards Reference, March 20, 2006, Version 3.7, JPL D-7669, Part 2.
- Planetary Science Data Dictionary Document, August 28, 2002, Planetary Data System (PDS), JPL D-7116, Rev E.
- Tools Survey, April 2005.

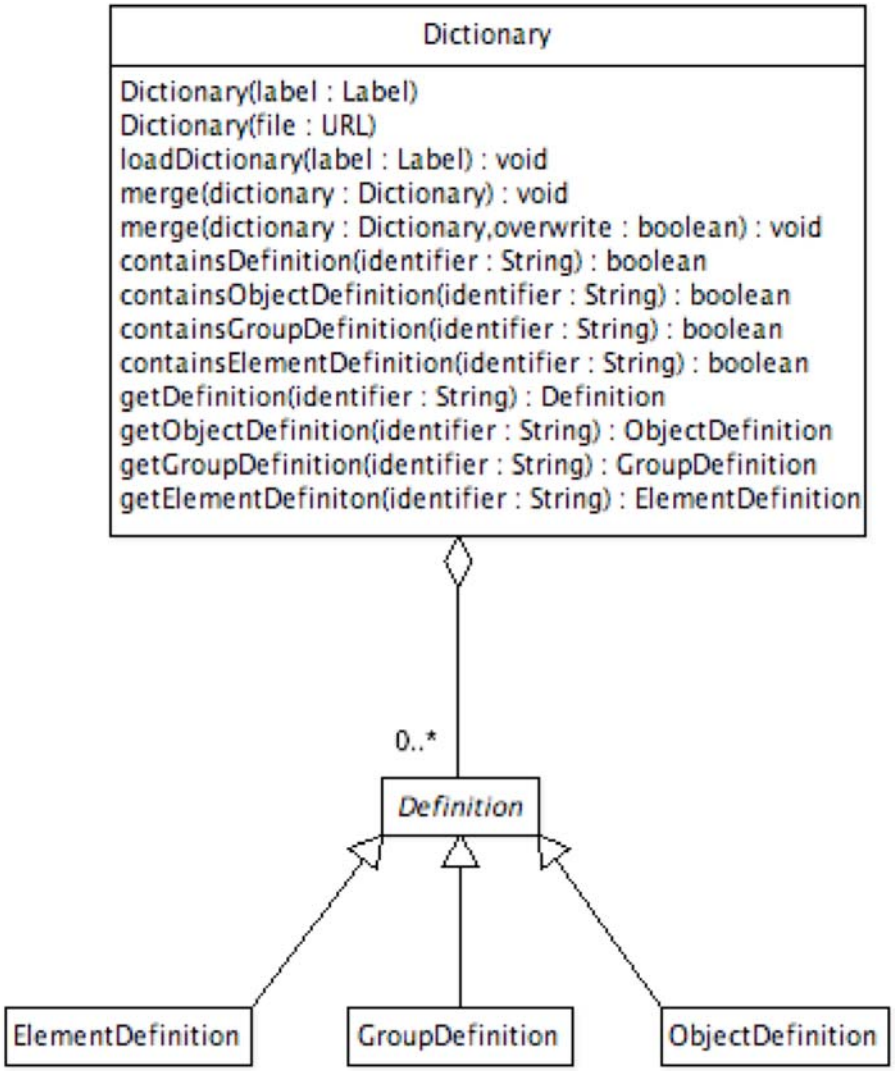
Closing Remarks, Action Items and Questions

Backup

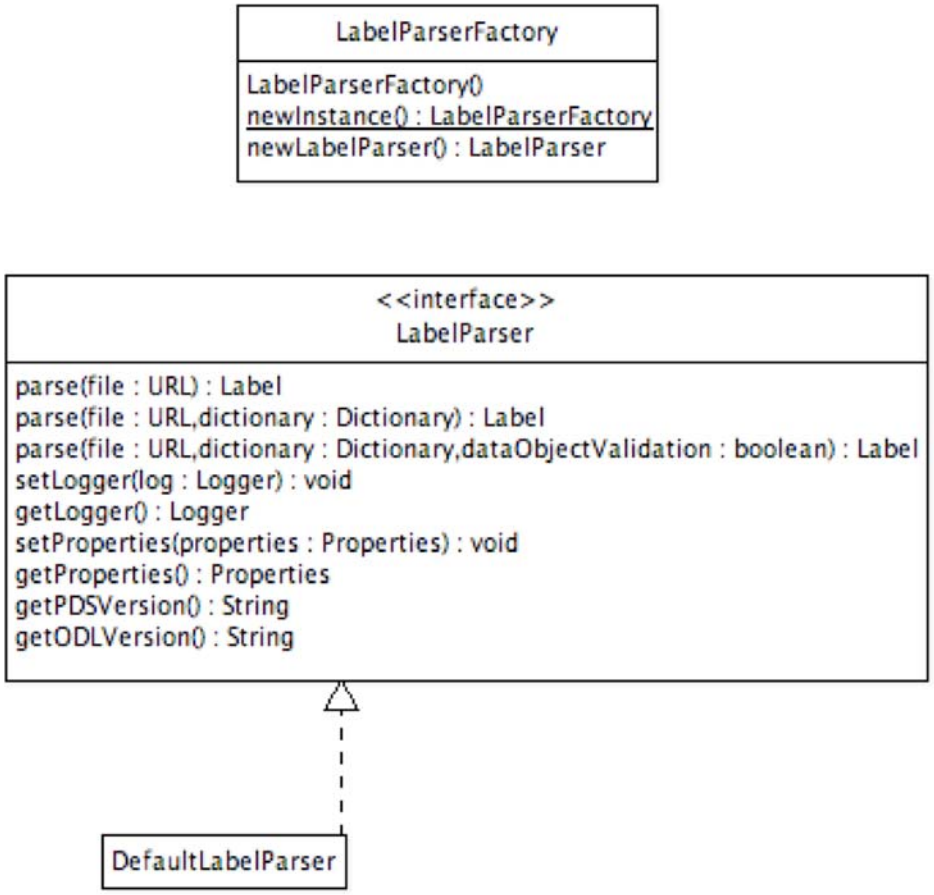
Software Design
Class Diagram - Label



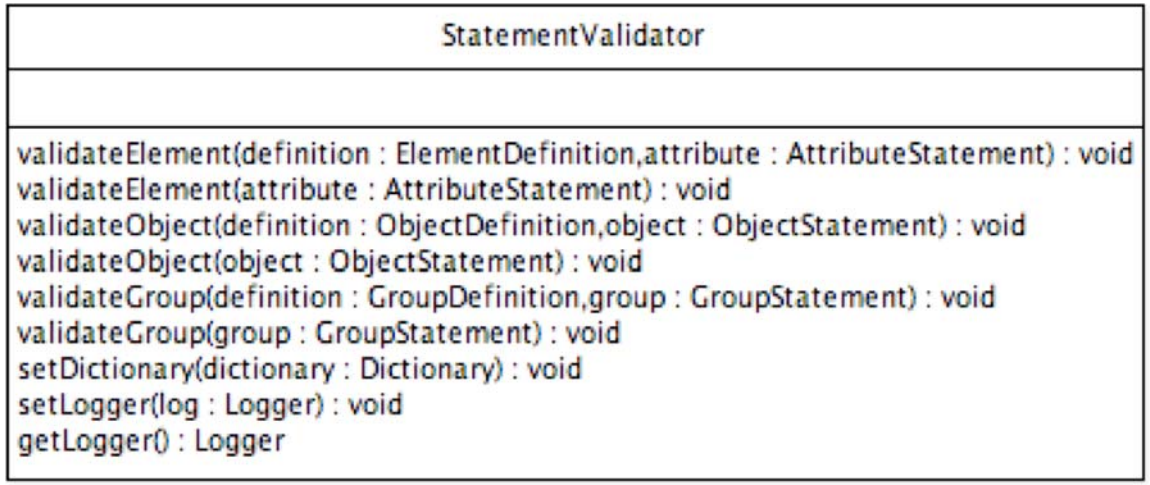
Software Design
Class Diagram - Dictionary



Software Design
Class Diagram - Parser



Software Design
Class Diagram - Statement Validation



Software Design
Class Diagram - Data Object Validation